 CMC, vol.60, no.1, pp.55-66, 2019

Efficient Analysis of Vertical Projection Histogram to Segment Arabic Handwritten Characters

Mamouni El Mamoun[1, *], Zennaki Mahmoud[1] and Sadouni Kaddour[1]

Abstract: The paper discusses the segmentation of words into characters, which is an essential task in the development process of character recognition systems, as poorly segmented characters will automatically be unrecognized. The segmentation of offline handwritten Arabic text poses a greater challenge because of its cursive nature and different writing styles. In this article, we propose a new approach to segment handwritten Arabic characters using an efficient analysis of the vertical projection histogram. Our approach was tested using a set of handwritten Arabic words from the IFN/ENIT database, and promising results were obtained.

Keywords: Segmentation, handwritten, Arabic text, vertical projection histogram.

1 Introduction

Writing recognition, a vast field of pattern recognition, is still a subject of intense research and experimentation. The problem is not yet fully solved, although in some applications where the vocabulary is limited or the font is unique or limited in number, we know how to obtain high rates. In addition, handwriting recognition is more complex than printed handwriting due to its extreme variability, variability of shapes, spacing between words and characters, line fluctuations.

The Latin language has received the greatest attention from researchers [Abhijit and Deeksha (2015); Tanzila, Amjad and Mohamed (2014)]. However, despite the number of people who speak Arabic, little research has been done on this language [Yasser (2013); Naz, Umar, Shirazi et al. (2016)], mainly because of the difficulty of segmenting words into letters. The segmentation step is an important step in the recognition process; this step is simple in the case of printed Latin text, but very difficult in the case of cursive writing (Arabic writing).

The complexity of the morphology of Arabic writing and its cursivity make it more difficult to segment words into characters. Several studies have been carried out by researchers based on the recognition of the entire word (global approach) without segmentation [Lawgali (2015)], and others assume that characters are already segmented to avoid the segmentation step [Lorigo and Govindaraju (2006); Khorsheed (2002)]. This

[1] Département Informatique Université des Sciences et de la Technologie d'Oran Mohamed Boudiaf USTO-MB, BP 1505 El M'naoeur, 31000, Oran, Algérie.

[*] Corresponding Author: Mamouni El Mamoun. Email: elmamoun.mamouni@univ-usto.dz.

step is a challenge for researchers and needs to be improved [Lawgali, Bouridane, Angelova et al. (2011)].

In fact, the development of a new segmentation algorithm is one of our objectives to make Arabic handwriting recognition more effective. The proposed algorithms perform a thorough analysis of the vertical projection histogram to extract the correct segmentation points.

This paper is organized as follows: handwritten Arabic characters are described in Section 2. In Section 3, we present some recent work on this subject. Section 4 describes the proposed approach in detail, while Section 5 discusses the results and their analysis. Section 6 summarizes the results of this work and draws conclusions.

2 Characteristics of Arabic script

Arabic language is a consonant script that uses a 28-letter alphabet. The shape of each letter depends on its position in the word, the same character can have up to four different shapes (isolated, beginning, middle and end), which increases the number of patterns, as illustrated in Tab. 1, 15 letters out of 28 have one or more dots, these dots can be above or below the character size, but never high and low simultaneously.

Table 1: Different shapes of an Arabic character.

Name	Isolated	Beginning	Middle	End
Alif	ا	ا	ا	ا
Baa	ب	بـ	ـبـ	ـب
Taa	ت	تـ	ـتـ	ـت
Thaa	ث	ثـ	ـثـ	ـث
Geem	ج	جـ	ـجـ	ـج
Hha	ح	حـ	ـحـ	ـح
Kha	خ	خـ	ـخـ	ـخ
Dal	د	د	ـد	ـد
Thal	ذ	ذ	ـذ	ـذ
Raa	ر	ر	ـر	ـر
Zain	ز	ز	ـز	ـز
Seen	س	سـ	ـسـ	ـس
Sheen	ش	شـ	ـشـ	ـش
Saad	ص	صـ	ـصـ	ـص
Dhad	ض	ضـ	ـضـ	ـض
Tta	ط	طـ	ـطـ	ـط
Zha	ظ	ظـ	ـظـ	ـظ
Ain	ع	عـ	ـعـ	ـع
Ghain	غ	غـ	ـغـ	ـغ
Faa	ف	فـ	ـفـ	ـف
Gaf	ق	قـ	ـقـ	ـق

Kaf	ك	كـ	ـكـ	ـك
Lam	ل	لـ	ـلـ	ـل
Meem	م	مـ	ـمـ	ـم
Noon	ن	نـ	ـنـ	ـن
Haa	ه	هـ	ـهـ	ـه
Waw	و	و	ـو	ـو
Yaa	ي	يـ	ـيـ	ـي

Arabic writing is written from right to left in a cursive manner in printed and handwritten characters; the characters of the same string are bound horizontally and sometimes vertically, as shown in Fig. 1.

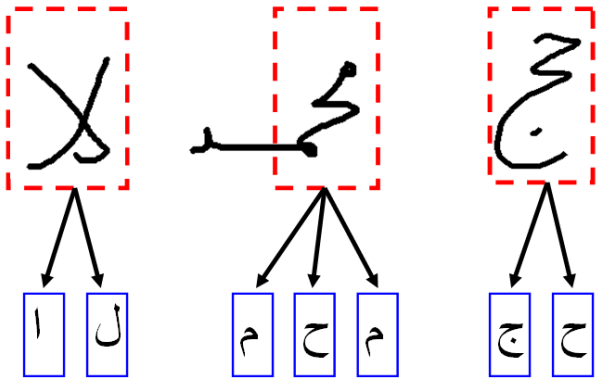

Figure 1: Vertical ligature

Some characters cannot be attached to their left, so they can only be isolated or in the final position; this gives, when they exist, words composed of one or more parts generally called PAW (Peace of Arabic Word) or even sub words, as shown in Fig. 2.

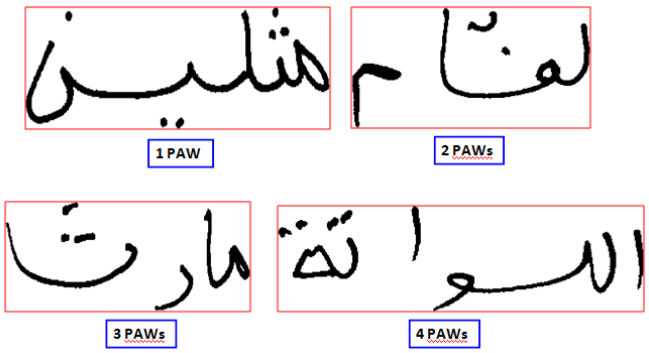

Figure2: Example of words composed of 1, 2, 3 and 4 PAWs

Vertical overlaps can occur through the intersection of descendants that extend horizontally below the baseline and the next secondary word, as shown in Fig. 3.

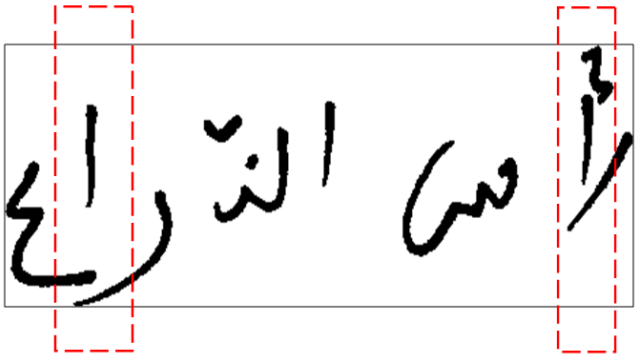

Figure 3: Example of overlap

This makes the problem of segmentation of Arabic words into characters and their recognition more difficult.

3 Related works

Several methods and algorithms have been developed to segment the handwritten text into characters.

As discussed in Gouda et al. [Gouda and Rashwan (2004)], the vertical projection and the baseline are used to segment a word into characters. The authors of Amin [Amin (1991)] select the weak points of each sub-word from the analysis of the vertical histogram, then the zero derivatives of the curvature contour are used to detect the convex dominant points.

In Syiam et al. [Syiam, Nazmy, Fahmy et al. (2006)], the k-means classification method was applied to the vertical histogram for word segmentation. This method increases the efficiency of the histogram by recognizing handwriting. The idea of the research presented in Yusra [Yusra (2013)] is to draw the contours of the sub-words, then the segmentation points are the points where the contour passes from a horizontal line to a vertical or curved line.

Lawgali et al. [Lawgali, Bouridane, Angelova et al. (2011)] first made the horizontal projection of the sub-word to determine the baseline. Then, the analysis of the vertical projection of the sub-word was performed to examine its distance from the baseline. Segmentation points that are far from the baseline are ignored.

Zaidi et al. [Zaidi, Khansa, Noorzaily, et al. (2009)] proposed a character segmentation algorithm based on the normalization of the histogram gradient sign and the sliding window technique for handwritten segmentation of Jawi characters. The authors of [Mohamed (2016)] have developed a segmentation algorithm that uses several techniques such as: spaces between words and sub-words, pen thickness, character width and text height.

These works are interesting, but they generally share a segmentation error greater than 15%, and if you add the error of the classification phase, these systems become

insufficient to recognize handwritten Arabic writing, so it is very interesting to go further to explore other techniques or to combine several methods to further improve the segmentation process.

4 Proposed approach

To segment a word into its characters, our approach consists of several steps. First, we generate the vertical projection histogram. Then, the word is segmented into sub-words followed by the extraction of the segmentation points. Subsequently, several operations are performed on the segmentation points to improve the position of these points or to delete someone. In the following, we present a detailed description of each step of the segmentation process of our approach.

4.1 Vertical projection histogram

The vertical projection represents the number of black pixels in each column of the image, defined by the vector V_j of size N as follows:

$$V_j = \sum_j P(i,j) \tag{1}$$

where $P(i,j)$ is a pixel of the binary image of the script and is either 0 or 1, i and j refer to indexes of the row and column.

4.2 Divide the word into its parts (sub-words)

Each word can be composed of one or more parts. We used the vertical projection vector V to determine the different parts of a word. The key idea is that when we find zero the value or a sequence of values of the projection vector ($Vj=0$), it means that the word can be divided into two sub-words in this position, as shown in Fig. 4.

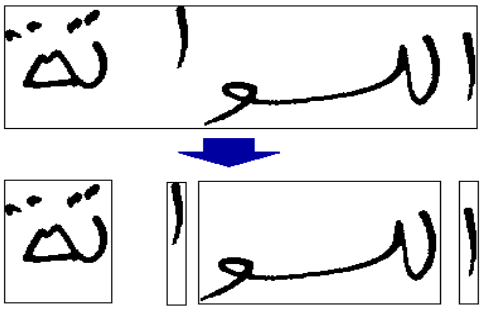

Figure 4: Divide word into its parts (sub-words)

4.3 Segmentation points extraction algorithm

We propose a new algorithm that uses the vertical projection vector (Eq. (1)) to extract the segmentation points. The algorithm has three parameters: Block size (Bs), Step size (Sp) and Threshold (T).

The value of the parameter Sp must be less than or equal to Bs, if the two parameters are equal there is no overlap between the block and the next one, as shown in Fig. 5.

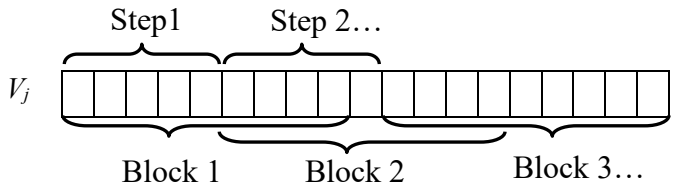

Figure 5: An example of vertical projection vector with Bs=9 and Sp=5

The concept of our algorithm is as follows: whenever the sum of the block values increases, any segmentation point is generated. If there is a significant decrease between the sums of the current values of the block compared to the previous block (above the threshold), there is a segmentation point. The following pseudo code describes our algorithm.

```
Input: Vertical projection histogram:  V[ ]
Output: Vector of segmentation points:  Seg[ ]
Set parameters: Bs, Sp and T
SumC=0; // Sum of current bloc values.
SumP=0; // Sum of preceding bloc values.
i=Sp; j=0; k=0; // i, j and k are integers
Begin
For (k=0; k<=Bs; k++)
        SumP= SumP +V[k];
End For
While (i <Image width - Bs) Do
For (k=i; k<=i+Bs; k++)
SumC=SumC+V[k];
End For
      If (SumP- SumC>T) then
          Seg[j]=i +Bs/2; //Keep the index of segmentation points
          j=j+1;
SumP=0;
Else
SumP=SumC;
End If
      i=i+Sp;
      SumC=0;
End while
End
```

In Fig. 6, we present an example of our algorithm application, the red vertical lines represent the position of the segmentation points.

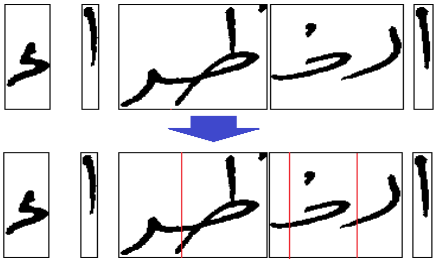

Figure 6: Examples of extraction of segmentation points

4.4 Improvement of segmentation points position

This operation consists of performing a local search around each segmentation point in order to find the best position for these points.

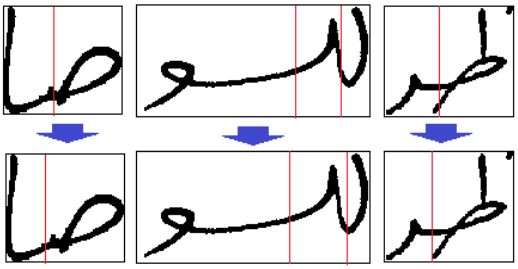

Figure 7: Improvement of segmentation points

The final segmentation point is the nearest point that has the smallest value in the vertical projection vector, around the initial point as shown in Fig. 7.

4.5 Baseline Detection

We performed the horizontal projection to determine the baseline. This operation deletes segmentation points far from the baseline as shown in Fig. 8.

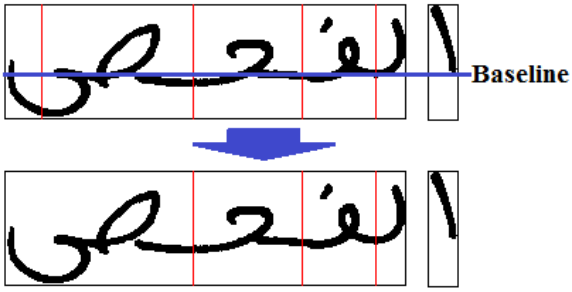

Figure 8: Removing segmentation point

4.6 Number of transitions

In each column of a segmentation point, we conducted a scan to find the number of times the pixel value changes state from 0 to 1 or from 1 to 0. The number of transitions is the total number of times the pixel state changes, a segmentation point is ignored if this number is greater than two, as shown in Fig. 9.

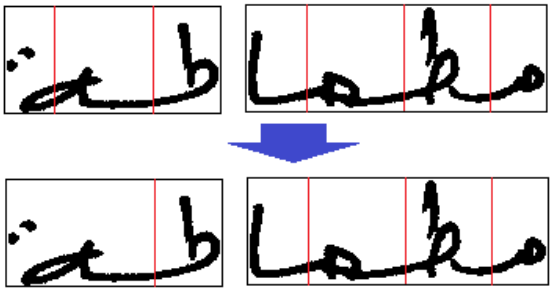

Figure 9: Example of number of transitions greater than two

5 Experimental results

To evaluate our approach, we used the IFN/ENIT database, which was developed in 2002 by the IFN (Institute of Communication Technologies in Germany) and ENIT (National Engineering School of Tunisia) in Tunisia. This is a database of Tunisian city names collected thanks to a contribution from 411 writers. Each of them wrote 60 names with their corresponding postal code.

The database contains 26459 city names in a lexicon of 946 cities, 115585 pseudo-words and 212211 characters. A full annotation of the city name images is made automatically, preceded by a manual check.

The algorithm developed was tested on a set of words from the IFN/ENIT database. The words have been carefully chosen to cover all forms of Arabic characters. The results obtained show that 89.5% of the segmentation points were extracted correctly, which is a very appreciable rate compared to the state of the art.

Our segmentation algorithm has three parameters and their values affect the results, in the following section we present a set of experiments to select the appropriate values of these parameters.

In the first experiment, we set the T threshold at 50 and the other two parameters are varied, the results obtained are presented in Fig. 10.

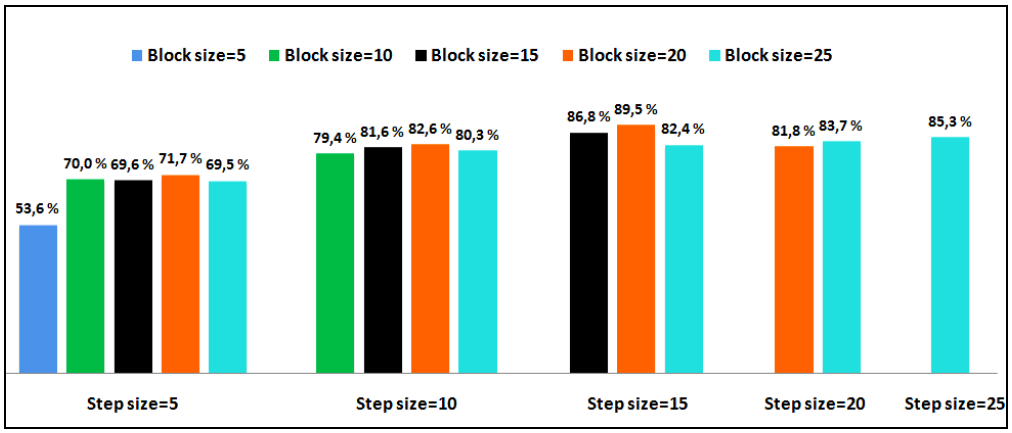

Figure 10: Character segmentation results with threshold T fixed at 50

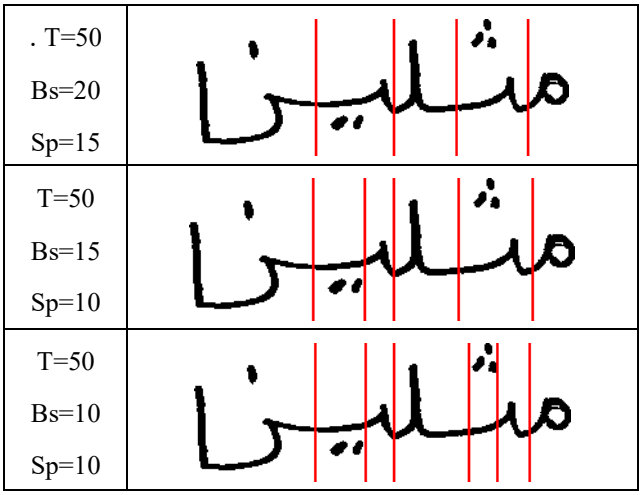

Figure 11: Example of over-segmentation

The best result obtained is 89.5% with Bs=20 and Sp=15. Segmentation errors are due to overlap or when we fall into over or under-segmentation. If the value of the parameters Bs and Sp is small, the error of over-segmentation increases and vice versa, as shown in Fig. 11.

In the second series of experiments, we set the block size and step size parameter (Bs=20, Sp=15), the results obtained by varying the threshold T are presented in Fig. 12.

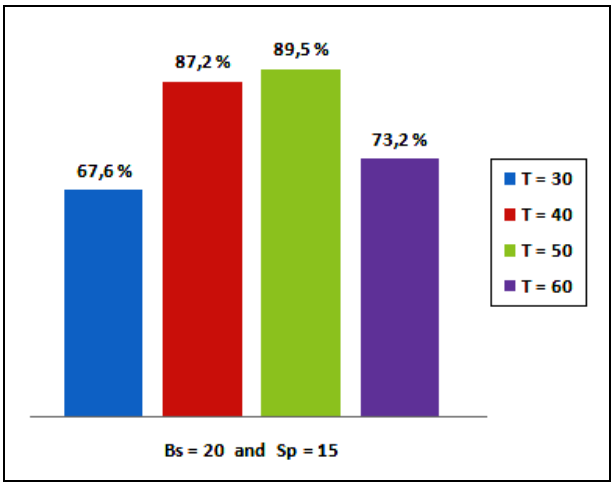

Figure 12: Character segmentation results with Bs=20 and Sp=15

The results show that the best value of the threshold is 50. Increasing the threshold more than 50 causes in many cases an under-segmentation and vice versa, as shown in Fig. 13.

T=50 Bs=20 Sp=15	
T=65 Bs=20 Sp=15	
T=80 Bs=20 Sp=15	

Figure 13: Example of under-segmentation

Finally, we present in Fig. 14 a comparison between the results obtained in the proposed algorithm and previous work.

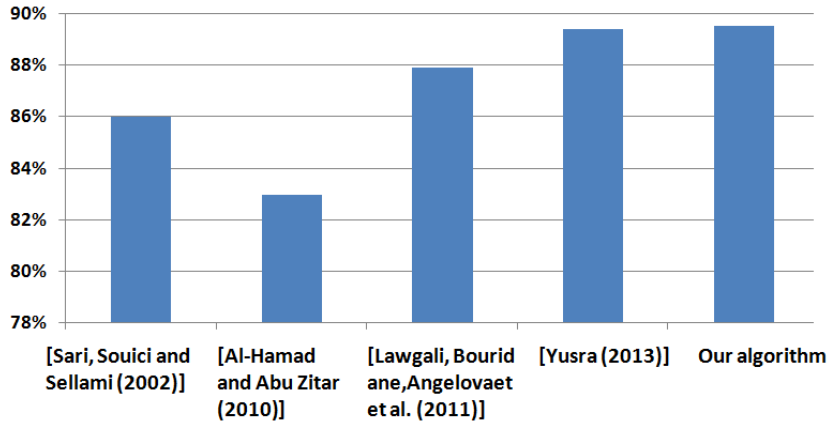

Figure 14: Comparison our results with previous works

6 Conclusion and future work

Recognizing characters after the segmentation process involves more challenges since segmentation introduces the most serious problem in the development of the cursive Arabic writing system.

The task of the segmentation word in its characters is more difficult than the segmentation that aims to extract lines and sub-words. In this work, a new segmentation algorithm is proposed based on a vertical projection histogram.

Our segmentation approach gives encouraging results, with an accuracy of 89.5%. The major problem that makes this task crucial is the problem of overlapping characters.

Therefore, on the basis of the promising findings presented in this paper, in future work, the segmentation algorithm will be improved by new research to solve the complex problem of overlapping characters.

References

Abhijit, J.; Deeksha, B. (2015): A survey on word segmentation method for handwritten documents. *International Journal of Science and Research*, vol. 4, no. 11, pp. 2319-7064.

Al-Hamad, H. A.; Abu Zitar, R. (2010): Development of an efficient neural-based segmentation technique for Arabic handwriting recognition. *Pattern Recognition*, vol. 43, no. 8, pp. 2773-2798.

Amin, A. (1991): Recognition of Arabic handprinted mathematical formulae. *Arabian Journal For Science and Engineering,* vol. 16, no. 4, pp. 532-542.

Gouda, A. M.; Rashwan, M. A. (2004): Segmentation of connected Arabic characters using hidden Markov models. *Proceedings of the IEEE International Conference on Computational Intelligence for Measurement Systems and Applications*, pp. 115-119.

Khorsheed, M. S. (2002): Off-line Arabic character recognition-a review. *Pattern Analysis & Applications*, vol. 5, no. 1, pp. 31-45.

Lawgali, A. (2015): A survey on Arabic character recognition. *International Journal of Signal Processing, Image Processing and Pattern Recognition,* vol. 8, no. 2, pp. 401-426.

Lawgali, A.; Bouridane, A.; Angelova, M.; Ghassemlooy, Z. (2011): Automatic segmentation for Arabic characters in handwriting documents. *Proceedings of the IEEE International Conference on Image Processing*, pp. 3529-3532.

Lorigo, L. M.; Govindaraju, V. (2006): Offline Arabic handwriting recognition: a survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 5, pp. 712-724.

Mohamed, A. F. (2016): An efficient segmentation algorithm for Arabic handwritten characters recognition system. *Proceedings of the IEEE International Conference on Mathematics and Computers in Sciences and in Industry*, pp. 172-177.

Naz, S.; Umar, A. I.; Shirazi, S. H.; Ahmed, S. B.; Razzak, M. I. et al. (2016): Segmentation techniques for recognition of Arabic-like scripts: a comprehensive survey. *Education and Information Technologies*, vol. 21, no. 5, pp. 1225-1241.

Sari, T.; Souici, L.; Sellami, M. (2002): Off-line handwritten Arabic character segmentation algorithm. *Proceedings of the IEEE International Workshop on Frontiers in Handwriting Recognition*, pp. 452-457.

Syiam, M.; Nazmy, T. M.; Fahmy, A. E.; Fathi, H.; Ali, K. (2006): Histogram clustering and hybrid classifier for handwritten Arabic characters recognition. *Proceedings of Pattern Recognition and Applications International Conference on Signal Processing*, pp. 44-49.

Tanzila, S.; Amjad, R.; Mohamed, E. B. (2014): Methods and strategies on off-line cursive touched characters segmentation: a directional review. *Artificial Intelligence Review,* vol. 42, no. 4, pp. 1047-1066.

Yasser, M. A. (2013): A survey on Arabic character segmentation. *International Journal on Document Analysis and Recognition*, vol. 16, no. 2, pp. 105-126.

Yusra, O. (2013): Segmentation algorithm for Arabic handwritten text based on contour analysis. *Proceedings of the IEEE International Conference on Computing, Electrical and Electronic Engineering*, pp. 447-452.

Zaidi, R.; Khansa, Z.; Noorzaily, M. N.; Rosli, S.; Mashkuri, Y. (2009): Offline handwritten Jawi character segmentation using histogram normalization and sliding window approach for hardware implementation. *Malaysian Journal of Computer Science*, vol. 22, no. 1, pp. 34-43.